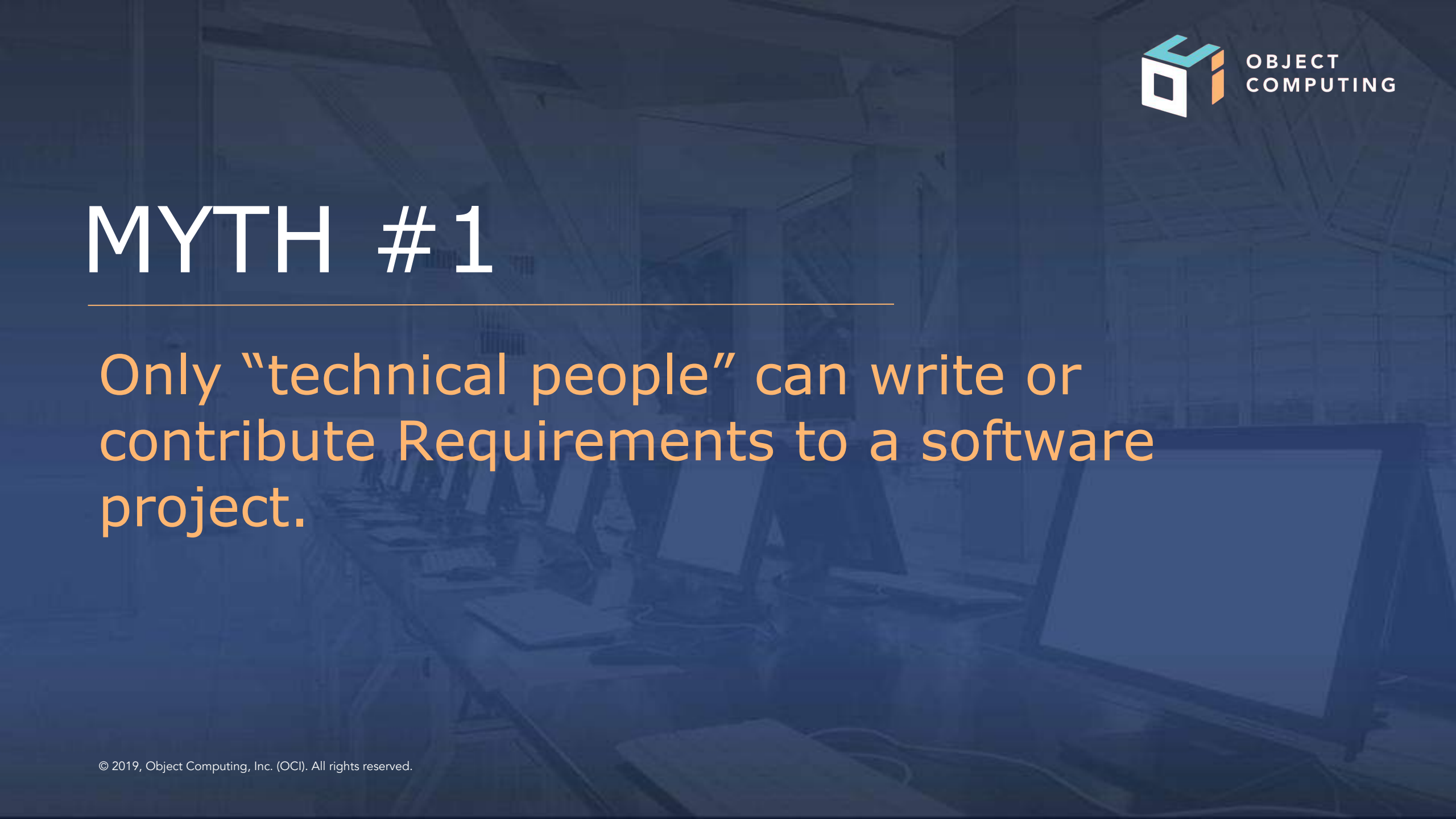
OBJECT
COMPUTING

Top 9 Myths Of Software Requirements Gathering

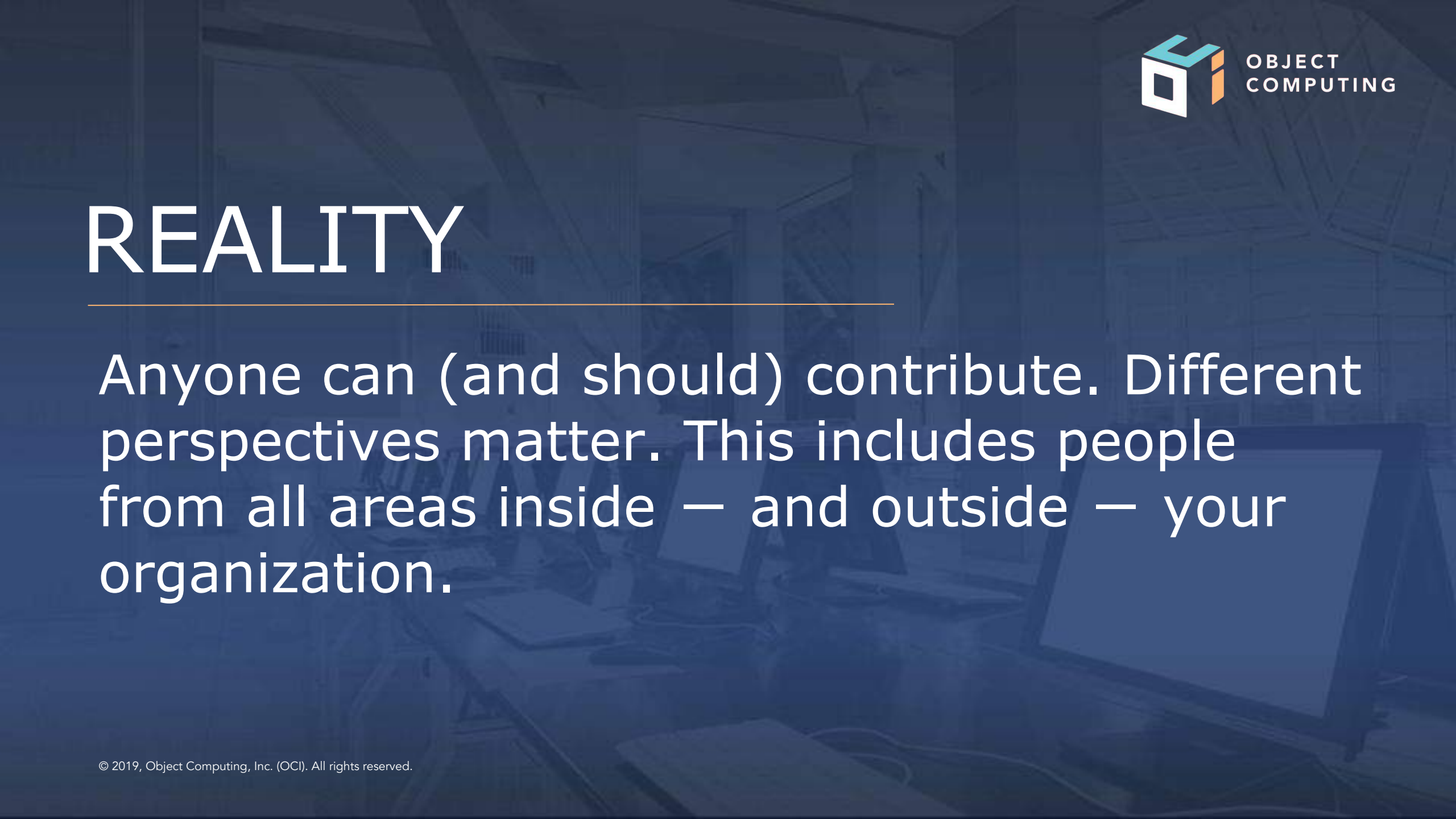
KEY CONSIDERATIONS

- Not a comprehensive course on Requirements Gathering.
- "Requirements Gathering" is one part of Requirements Management.
- Seek advice if you have regulatory or compliance needs.
- There are many requirements methodologies; any of them is (almost always) better than none at all.
- Experience makes a difference. You will improve over time. It helps to work with experts. Take advantage of experience.
- Keep it simple. Requirements are not the objective, just a way to get there.



MYTH #1

Only “technical people” can write or contribute Requirements to a software project.



REALITY

Anyone can (and should) contribute. Different perspectives matter. This includes people from all areas inside — and outside — your organization.

Contributing Requirements

- Cast a wide net to find as many potential stakeholders as possible.
- Balancing Act: Encouraging participation while minimizing overzealous input.
- Use your backlog to table unnecessary discussions (“We’ll add it to the backlog and discuss it in future grooming sessions.”)
- Avoid “Secret Requirements Meetings”.

Requirements Contributors*

- Business Leadership
- Legal
- IT
  - Development
  - Security
  - Analytics
  - Operations
- End Customers
- End-Users
- Operations
- Research and Development
- Engineering & Maintenance
- Health and Safety
- Audit and Regulatory Compliance
- Human Resources
- Independent Third Parties (outside experts)

* These are examples of contributing roles. Your contributors may differ.

MYTH #2

Requirements are just Technical Specifications.

OBJECT
COMPUTING

REALITY

Requirements should represent a solution to a business problem. Understanding the business problem is critical to success.

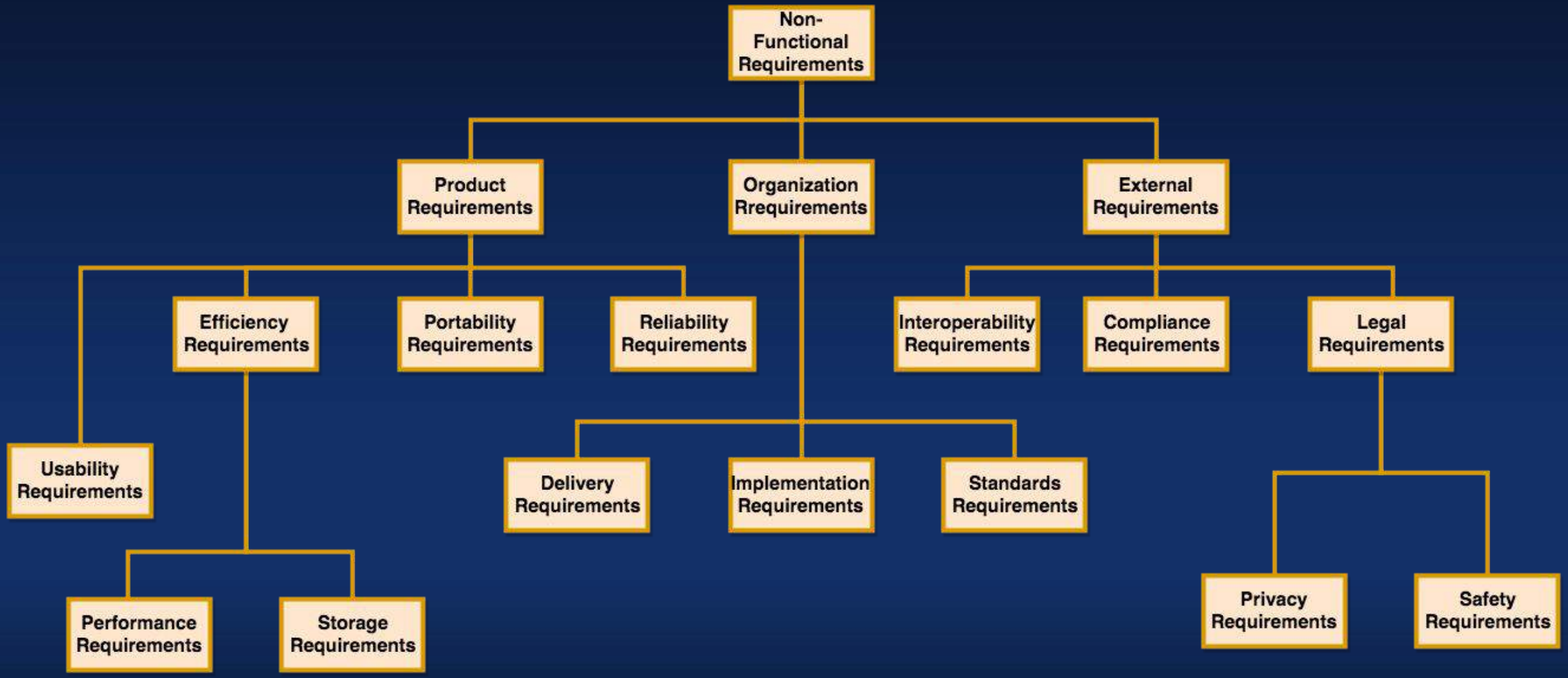
Requirements are a collection of business features.

Requirements – Requirement Types

OBJECT COMPUTING

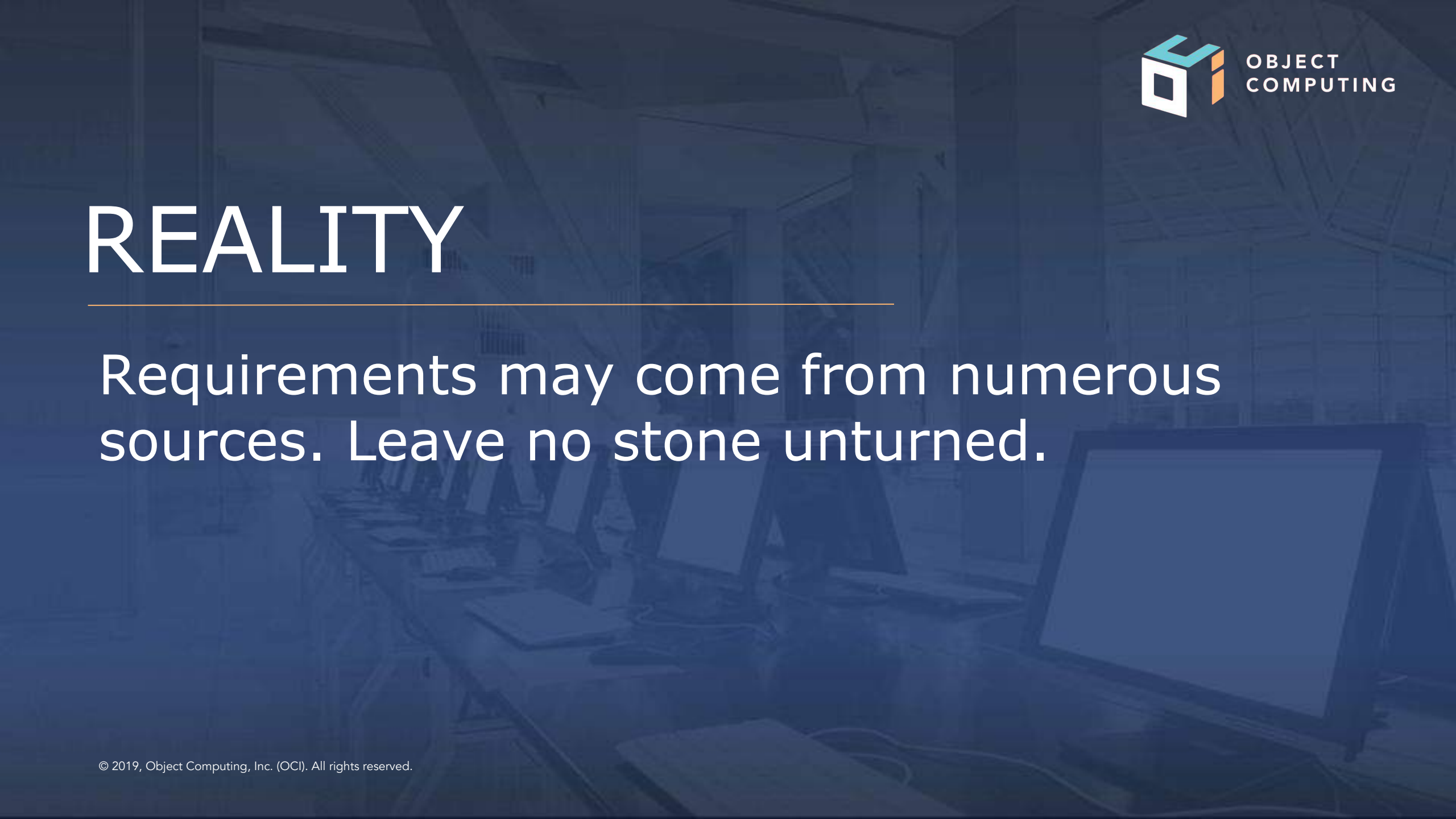
- Functional – Things the product or application must do
  - "Business capabilities"
  - What features does it have?
- Non-Functional – Qualities the product or application must have
  - Does it need to be fast?
  - Does it have to be globally available?
  - How secure must it be?
  - What platform does it work on?

Requirements – Non-Functional Requirements



MYTH #3

Requirements only come from the business stakeholders who pay for the product or application.



REALITY

Requirements may come from numerous sources. Leave no stone unturned.

(Not All) Requirements Gathering Sources

OBJECT COMPUTING

- One-on-One Interviews
- Group Interviews and Focus Groups
- Structured Workshops
- Brainstorming
- Surveys/Questionnaires
- Prototyping
- Use Case Diagrams
- Competitor Analysis
- User/Consumer Observation
- Existing Documentation
- Existing Solutions and Tools
- RFP/RFD Documents

OBJECT
COMPUTING

MYTH #4

If you work for a large organization, you need sophisticated tools and a complex requirements-management methodology.

REALITY

Your mission, your organizational culture, and the intended outcome should be the prime considerations in selecting requirements-management tools and methodology.

Requirements Tools - Avoid Pitfalls

OBJECT COMPUTING

- Don’t let a Requirements Management tool become a burden.
- Make sure Stakeholders have access to the tool.
- Capture critical raw data (electronic info, paper docs, whiteboard, audio, etc.)
- Spreadsheets don’t work.

Requirements Management Tool - Features

OBJECT COMPUTING

- Store Requirement Statements
- Organize Statements: Category, Type and other attributes
- Attach or reference external artifacts (documents, pictures, etc.)
- Versioning
- Prioritization
- Consistency Enforcement
- Identify Requirement Gaps (e.g. “to be defined” areas)
- Traceability
- API Interfaces
- Interface with Testing tools

MYTH #5

Once Requirements are handed off to the Development Team, the implementation is up to them.

REALITY

If you gather Requirements for the Development Team, then you are part of the Team.

Requirements Gathering – Part of the Dev Team

OBJECT COMPUTING

- Development team success (or failure) is yours.
- The Over-the-Wall approach is full of risk, disappointment and re-work.
- Stay Engaged.
- Collaborate.

Critical Data – Requirements Details

OBJECT COMPUTING

Good Requirements follow a clear, consistent criteria

S.M.A.R.T Requirements Method*

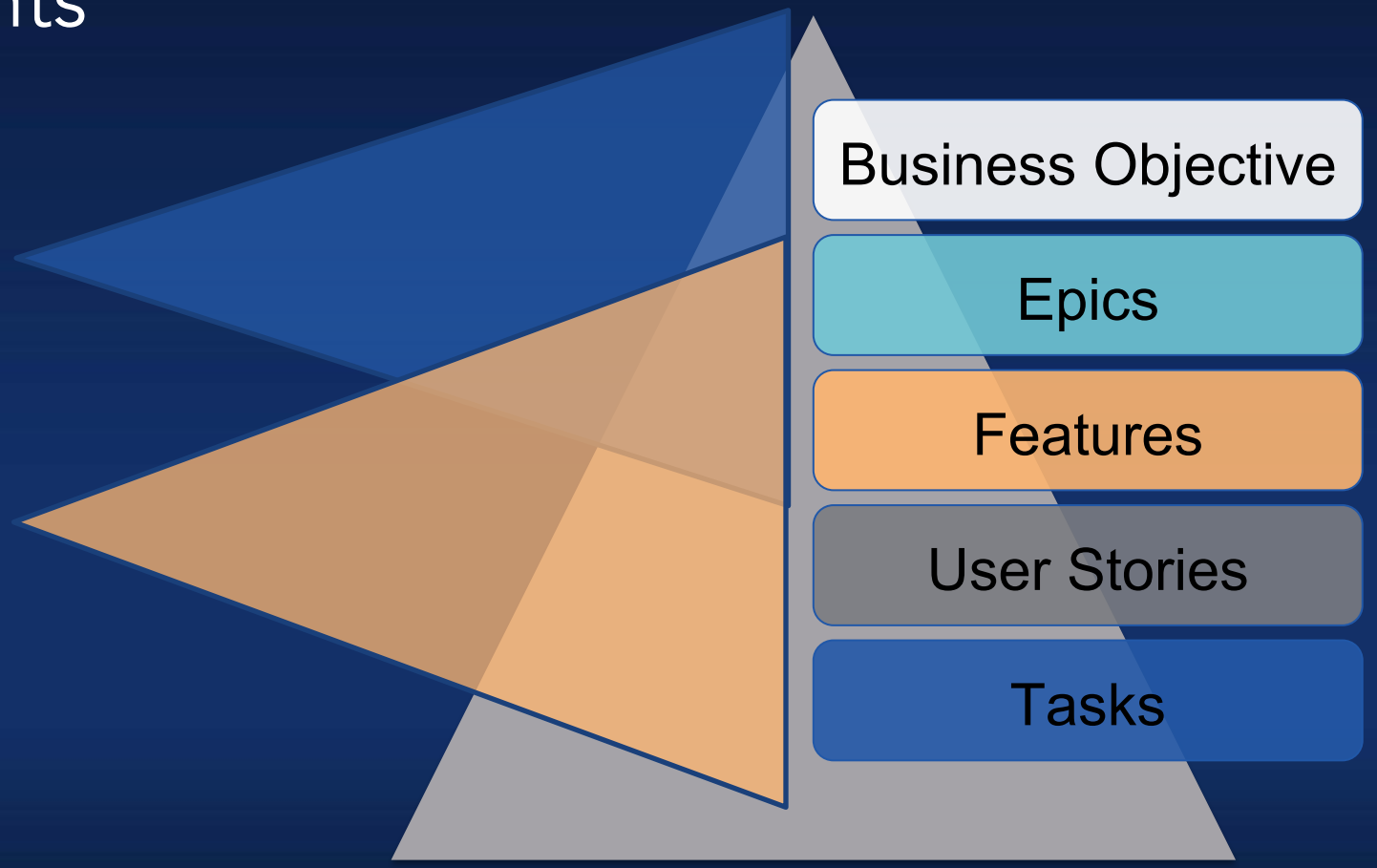
- S - Specific
- M - Measureable
- A - Attainable (Actionable, Achievable, Appropriate)
- R - Realistic
- T - Time Bound (Timely, Traceable)

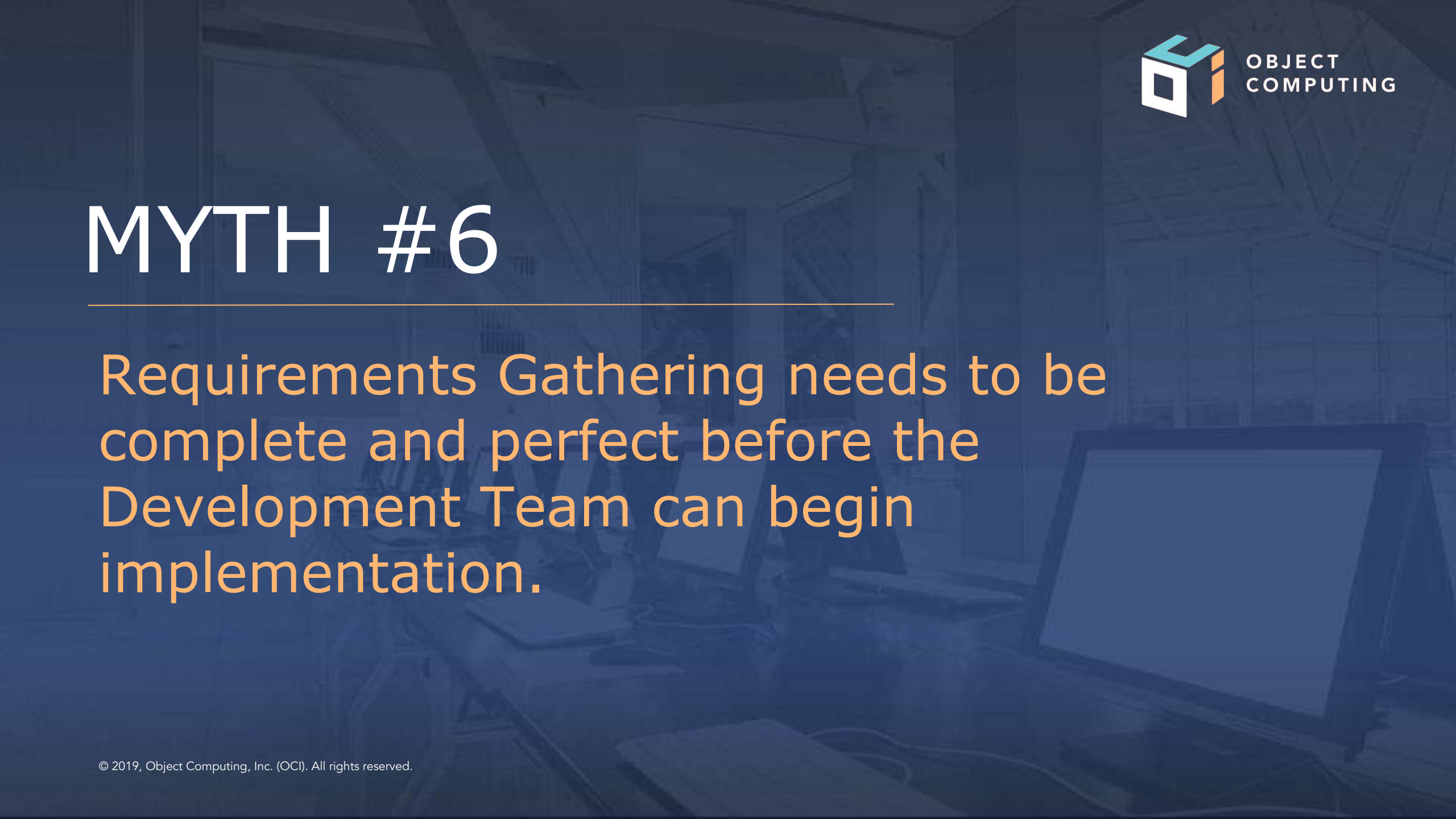
* There are other methods. Use one method consistently!

Critical Data – Requirements Hierarchy

OBJECT COMPUTING

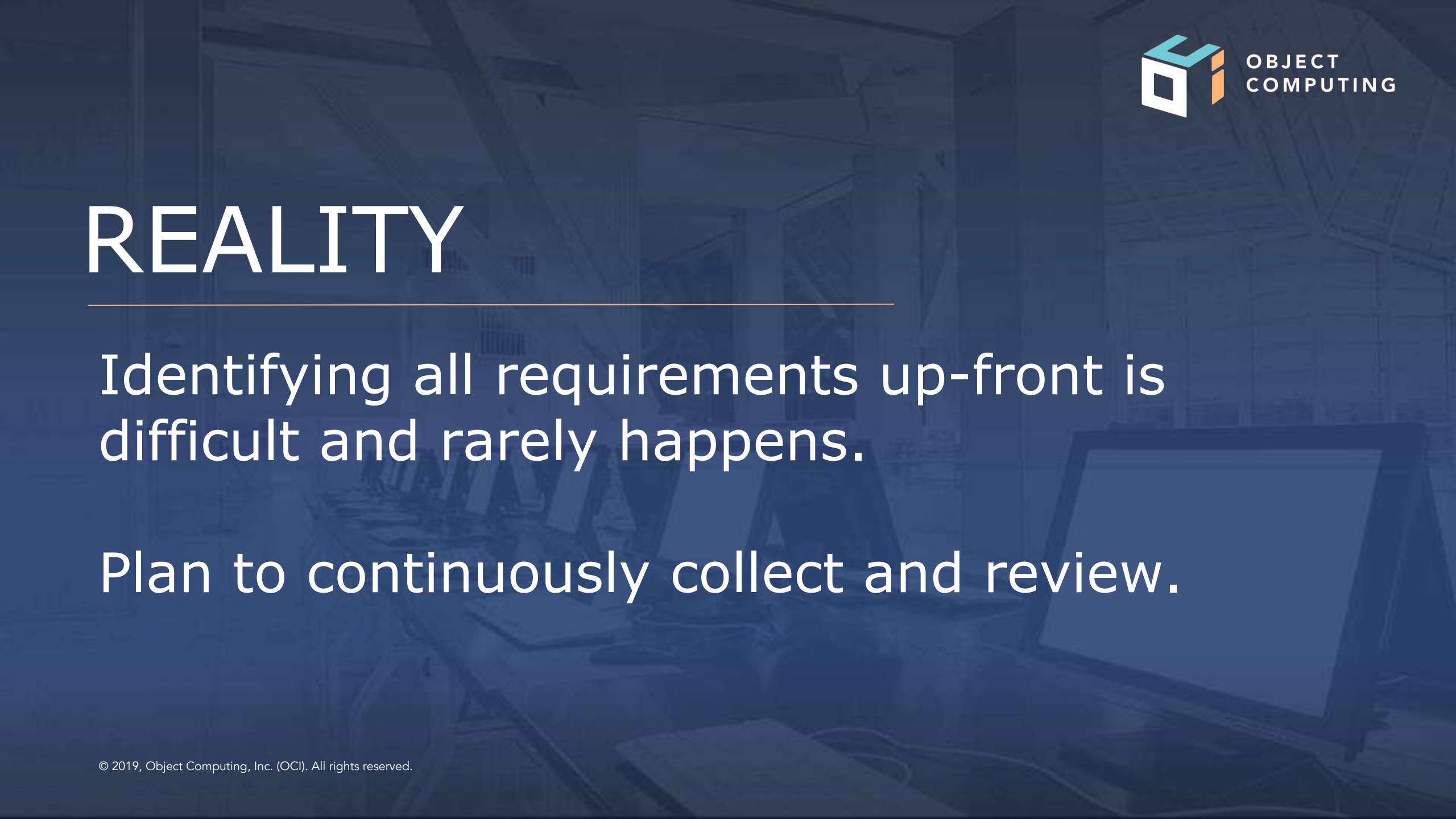
Pyramid of Requirements





MYTH #6

Requirements Gathering needs to be complete and perfect before the Development Team can begin implementation.



REALITY

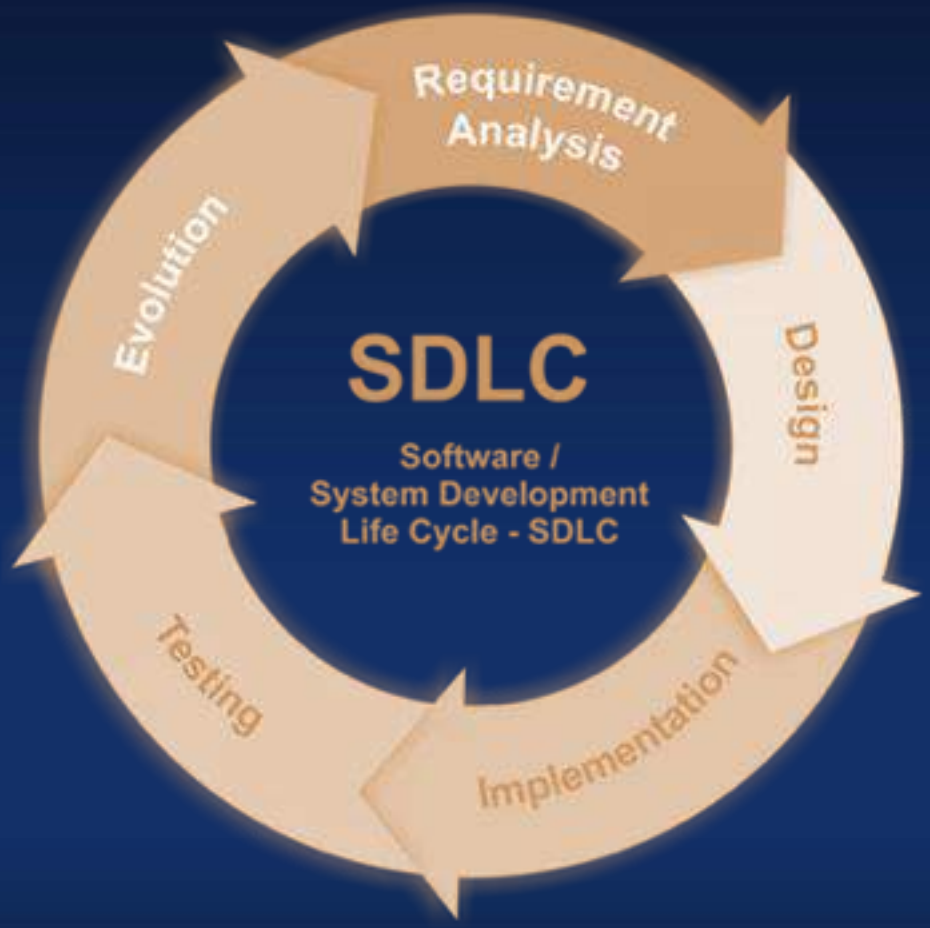
Identifying all requirements up-front is difficult and rarely happens.

Plan to continuously collect and review.

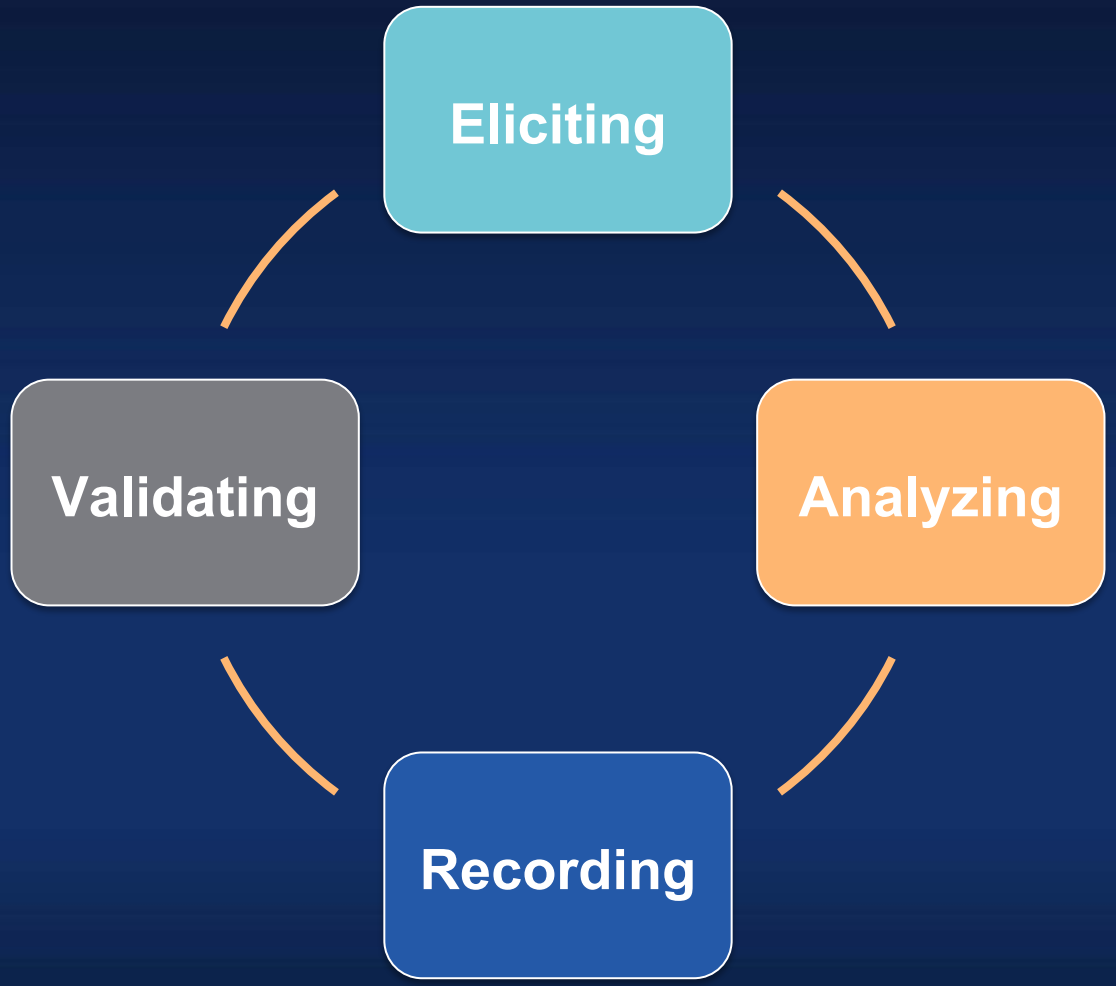
Requirements Gathering – Part of the SDLC

OBJECT COMPUTING

- Requirements Gathering is part of the the Software Development Life Cycle and should be repeated continuously.
- Take advantage of what is learned in each cycle.
- Review results and feedback. Adjust and improve your product or application.



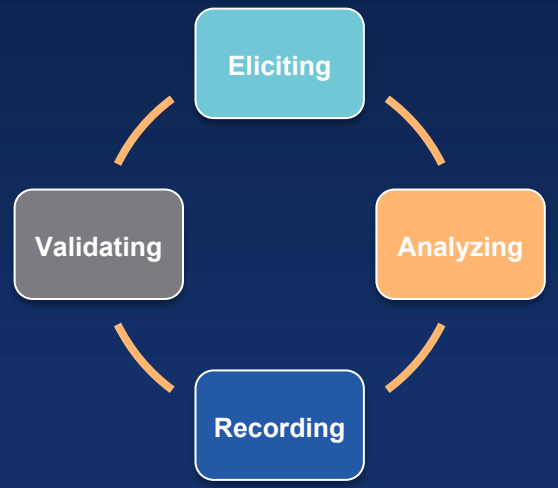
Requirements Gathering - Key Steps

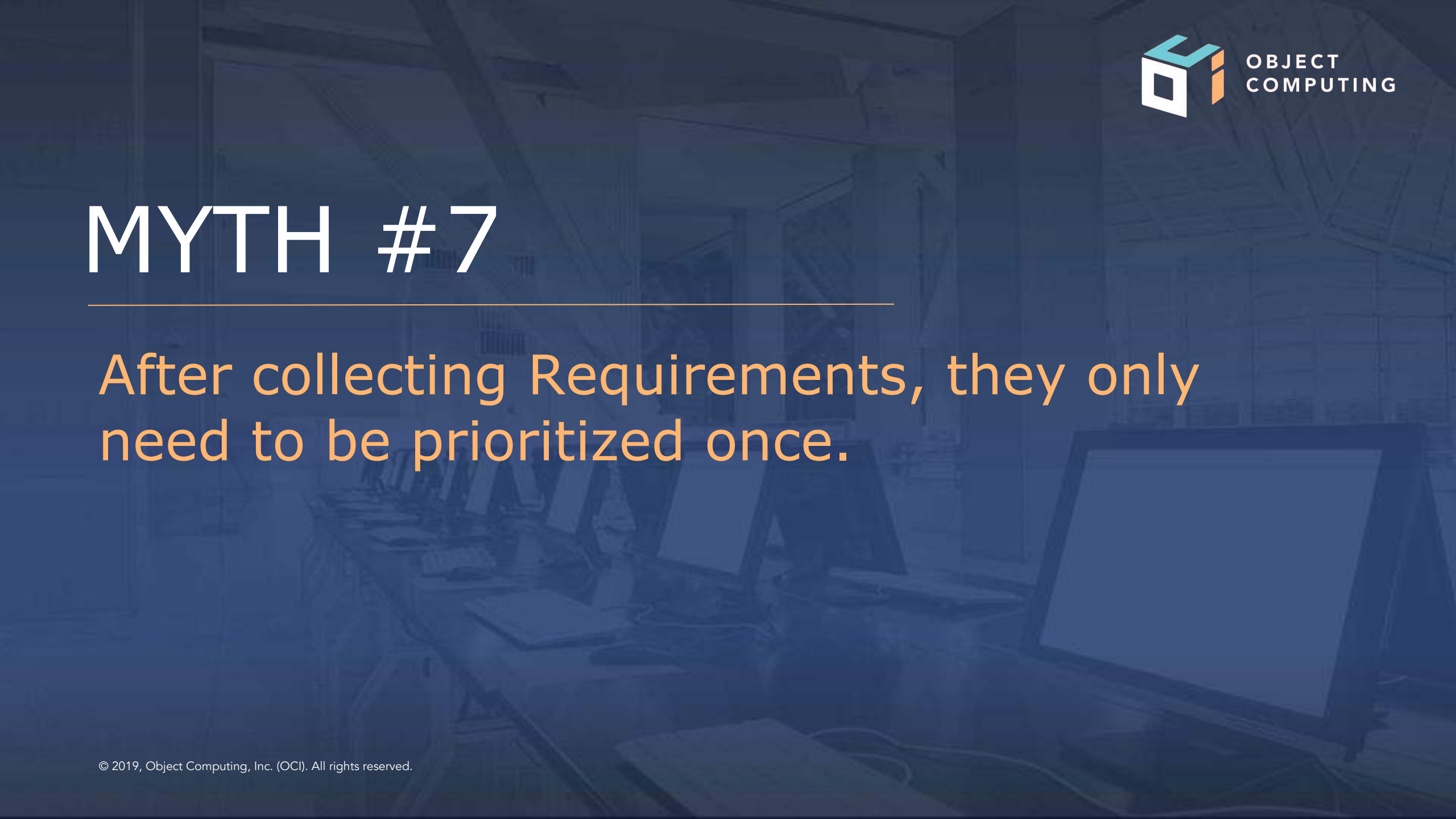


Requirements Gathering - Key Steps

There are four steps that need to be done in Requirements Gathering:

1. **Eliciting**– Ask questions and listen to the answers
2. **Analyzing**– Organize and consolidate requirements
3. **Recording** – Document and store requirements
4. **Validating–** Confirm requirements meet business need





MYTH #7

After collecting Requirements, they only need to be prioritized once.

REALITY

The marketplace is ever-changing. What customers (or end-users!) want or need can evolve over time. Adapting to change is critical for business success.

Evaluating priorities of remaining work after every cycle can greatly improve the overall quality of a product or application.

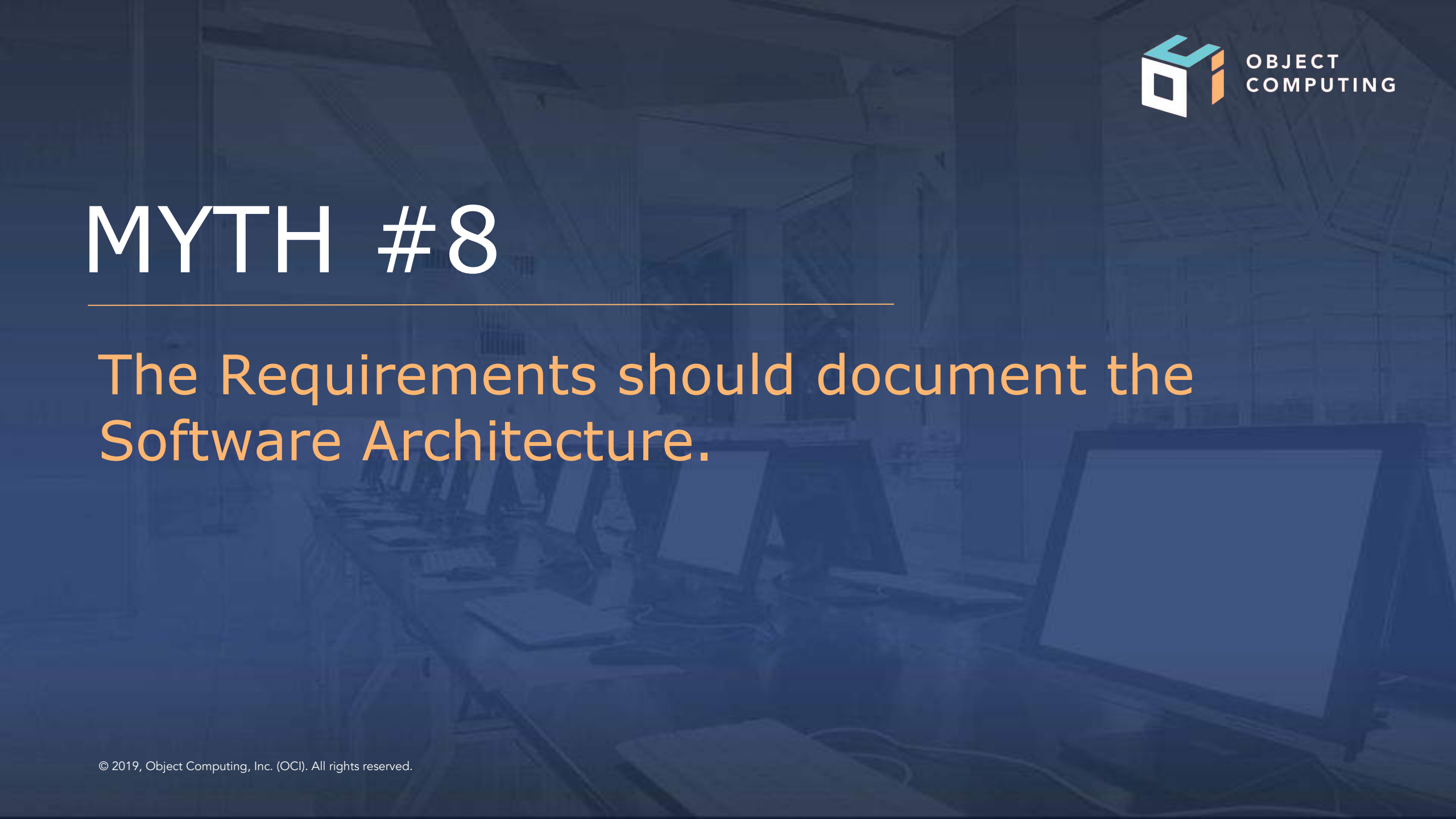
Critical Data – Requirements Prioritization

OBJECT COMPUTING

MoSCoW method of Prioritization

- M - Must Have
- S - Should Have
- C - Could Have
- W - Won't Have*

* Variant of W: "Wish to Have"



MYTH #8

The Requirements should document the Software Architecture.

REALITY

Software Architecture and Software Requirements are not the same.

Requirements should describe the business solution.

Software Architecture in Requirements

OBJECT COMPUTING

- Requirements are “What needs to be done.”
- Architecture is “How it is done.”
- Sometimes Technical Requirements are called “Constraints”.

Software Architecture in Requirements

OBJECT COMPUTING

- Constraints are just a form of Non-Functional Requirement.
- Minimize Constraints to maximize Development Team options.
- If the “Business Solution” is Technical, then so should be the Requirements.

MYTH #9

Democracy is the best way to identify and prioritize Requirements.

REALITY

Majority rule is probably the least productive way to manage requirements. Consensus is a close second.

Requirements – Decision Making Pitfalls

OBJECT COMPUTING

- Majority Rule decision making: Loudest voices generally prevail (or get the most attention) in Group Interviews or Focus Groups.
  - Use other means to gather quiet (but important) viewpoints – One-on-One Interviews, Surveys, Observation, etc.
- Majority rule decision making may not align with the Business objectives.
- Consensus -based decision making can also hide important viewpoints – sometimes people do not speak up.

Requirements – Decision Making – Best Practices

OBJECT COMPUTING

- Identify a facilitator for the Requirements Gathering effort.
- Identify a final decision maker (Product Owner) – a separate role from facilitator.
- Include many voices in the process.
- Try to keep prioritization as a separate effort.
- Take especially difficult decisions out of public view; Communicate decision publicly to close the matter.
- Strive for consensus but know when Product Owner will make the decision.
- Requirements Gathering is rarely ever "complete" – even after a product is deployed or shipped.

OBJECT COMPUTING

Recap

Top 9 Practices and Tips Of Software Requirements Gathering

OBJECT COMPUTING

RECAP

1. Anyone can (and should) contribute to the Requirements Gathering process.
2. Requirements are a collection of Business Features. Leave the technical details to your technical experts.
3. Requirements come from many sources, both people and documents. Look high and low.
4. Pick tools that make gathering and managing requirements easier and more efficient. Don’t overdo it (or underdo it). Don’t use spreadsheets.

OBJECT COMPUTING

RECAP

5. Good Requirements are S.M.A.R.T. (Specific, Measurable, Attainable, Realistic, Time Bound)

6. Stay engaged with the Development Team during the entire SDLC. Help improve and refine Requirements continuously.

7. Use the MoSCoW method to prioritize your Requirements. (Must Have, Should Have, Could Have, Won't Have)

8. Requirements are "What needs to be done." Architecture is "How it is done."

9. Have a Product Owner to streamline decision making.

LEARN MORE ABOUT OCI EVENTS AND TRAINING

Events:

- objectcomputing.com/events

Training:

- objectcomputing.com/training
- grailstraining.com
- micronauttraining.com

Or email info@ocitraining.com to schedule a custom training program for your team online, on site, or in our state-of-the-art, Midwest training lab.

OBJECT COMPUTING

CONNECT WITH US

1+ (314) 579-0066

@objectcomputing

objectcomputing.com